Программирование на языке Python

Тема 2: Области видимости переменных. Вложенные функции

Понятие вложенной функции

```
def func1():
    print('Создаем вложенную функцию..')

    def func2():
        print('Выполняется вложенная функция..')

    func2()
    print('Завершилась основная функция.')

func1()
```

```
Создаем вложенную функцию..
Выполняется вложенная функция..
Завершилась основная функция.

Process finished with exit code 0
```

Локальные, глобальные и нелокальные переменные

```
name = 'Олег'


def print_hello():
    print('Привет,', name)


print_hello()
```

```
Привет, Олег

Process finished with exit code 0
```

Локальные и глобальные переменные

Глобальные переменные (они так называются потому, что были созданы в глобальном поле видимости) на то и глобальные, что доступны везде: в функциях, во вложенных функциях, вне каких-либо функций и так далее.

Локальные и глобальные переменные

```
name = 'Олег'


def print_hello():
    name = 'Артем'
    print('Привет,', name)


print_hello()
print(name)  # изменится ли глобальная переменная?
```

Локальные и глобальные переменные

```
name = 'Олег'

def print_hello():
    name = 'Артем'
    print('Привет,', name)
```

Локальная переменная **name** затенила глобальную переменную с таким же именем!

```
print_hello()
print(name)
```

```
Привет, Артем
Олег

Process finished with exit code 0
```

Изменение глобальной переменной из функции

```
counter = 0


def func():
    counter += 1
    print('Функция была вызвана', counter, 'раз')


func()
func()
func()
```

Изменение глобальной переменной из функции

```
counter = 0


def func():
    counter += 1
    print('Функция была вызвана', counter, 'раз')


func()
func()
func()
```

```
Traceback (most recent call last):
  File "D:/YandexDisk/Programming/Python/PyCharm WorkSpace/test/main.py", line 9, in <module>
    func()
  File "D:/YandexDisk/Programming/Python/PyCharm WorkSpace/test/main.py", line 5, in func
    counter += 1
UnboundLocalError: local variable 'counter' referenced before assignment

Process finished with exit code 1
```

В чем проблема?

```
counter = 0


def func():
    counter = counter + 1  # изменили инкремент
    print('Функция была вызвана', counter,
'раз')


func()
func()
func()
```

Инструкция global

```
counter = 0


def func():
    global counter
    counter = counter + 1
    print('Функция была вызвана', counter, 'раз')


func()
func()
func()
```

```
Функция была вызвана 1 раз
Функция была вызвана 2 раз
Функция была вызвана 3 раз

Process finished with exit code 0
```

Инструкция global

```
computer_performance = 0
computer_weight = 0

computer_type = 0  # 0 - стационарный компьютер, 1 - ноутбук


def init_computer():
    global computer_performance, computer_weight
    if computer_type == 0:
        computer_performance = 100
        computer_weight = 10
    elif computer_type == 1:
        computer_performance = 75
        computer_weight = 3


init_computer()
print(computer_performance)
print(computer_weight)
```

```
100
10

Process finished with exit code 0
```

Локальные переменные и аргументы функции

- Аргументы функции являются локальными переменными и могут затенять глобальные переменные

```
place = 'Сургут'


def print_invite(name, place):
    print('Привет, ', name, '!', sep='')
    print('Приглашаем вас посетить наш чудесный город:', place)


print_invite('Маша', 'Сухой Лог')
```

```
Привет, Маша!
Приглашаем вас посетить наш чудесный город: Сухой Лог

Process finished with exit code 0
```

Нелокальные переменные

```
var1 = 0


def func1():
    var2 = 1

    def func2():
        var3 = 3
        print(var1, var2, var3)

    func2()


func1()
```

Как изменить переменную var1 из функции func2?

Нелокальные переменные

```
var1 = 0


def func1():
    var2 = 1

    def func2():
        global var1
        var3 = 3
        var1 = 4
        print(var1, var2, var3)

    func2()
    print('var1 после вызова func2:', var1)


func1()
```

```
4 1 3
var1 после вызова func2: 4

Process finished with exit code 0
```

Нелокальные переменные

```
var1 = 0


def func1():
    var2 = 1

    def func2():
        global var1
        var3 = 3
        var1 = 4
        print(var1, var2, var3)

    func2()
    print('var1 после вызова func2:', var1)


func1()
```

```
4 1 3
var1 после вызова func2: 4

Process finished with exit code 0
```

А как тогда изменить переменную var2 из функции func2?

Нелокальные переменные

```
var1 = 0


def func1():
    var2 = 1

    def func2():
        global var1
        nonlocal var2
        var3 = 3
        var2 = 5
        var1 = 4
        print(var1, var2, var3)

    func2()
    print('var1 после вызова func2:', var1)
    print('var2 после вызова func2:', var2)


func1()
```

```
4 5 3
var1 после вызова func2: 4
var2 после вызова func2: 5

Process finished with exit code 0
```

Правило LEGB

LEGB – local, enclosing, global, built-in – порядок поиска переменных при обращении к ним.

При желании изменить переменную:

- Если переменная в поле L (local), то действий не требуется
- Если переменная в поле E (enclosing), то объявить ее, как nonlocal
- Если переменная в поле G (global) или B (built-in), то объявить ее, как global